Dokumentaatio: Kello (Clock)

Analoginen kello

Aloitin projektin tekemällä kuvat kellosta ja sen viisareista Photoshopissa. Siirsin kuvat Unityyn, asetin jokaisen position nolliin, skaalasin ne sopivan kokoisiksi, järjestin oikein ja asetin ne tyhjän objektin lapsiksi (Kello). Jokaista viisaria varten täytyi luoda uusi tyhjäobjekti ja asettaa se viisarin päähän ankkuriksi, jotta viisarit toimisivat niin kuin niiden pitäisi.

Script

Loin Kello-scriptin, joka tulee hallitsemaan kellon toimintaa ja asetin sen kelloController-objektille. Scriptin sisälle kirjoitin kolme peliobjekti-muuttujaa: sViisari, mViisari ja tViisari ja asetin editorissa niihin oikeat objektit. Kellon toiminnan toteutin käyttämällä DateTime-tyyppiä, joka antaa järjestelmän päivämäärän ja ajan. ”DateTime.Now” antaa ajan reaaliajassa ja asetin sen muuttujaan ”aika”. Toimiakseen DateTime tarvitsee oikean namespacen (System), jota scripti käyttää. Viisareiden liikehdintää varten täytyi luoda kolme uutta muuttujaa: sAste, mAste ja tAste, jotka asettavat viisarit oikeaan kulmaan. Viisarit oikeaan kulmaan asettaa ”Viisari.transform.eulerAngles”, mutta ”Viisari.transform.localRotation = Quaternion.Euler(...)” käy myös, mutta siitä tulee pidempi lause.

Astelasku

otetaan DateTime-muuttujasta ”aika” halutut aikayksiköt, esim. sekuntiviisarin tilanteessa otetaan sekunnit sekä millisekunnit, jotta viisarien liikehdintä ei olisi niin robottimaista. Jokaisen luvun pitää olla tyyppiä float, jotta scripti ymmärtää lisätä desimaalit. Pehmentävä aikayksikkö eli aiemmassa esimerkissä millisekunnit tulee jakaa 1000:lla, jotta se antaa oikean arvon sekunnille, kun se lisätään sekuntiarvoon. Sekunnit ja millisekunnit jaetaan sitten 60:llä, joka kerrotaan -360 asteella, jotta saadaan oikea kulma ja suunta viisareille.

Tämä toteutetaan jokaiselle viisarille hieman eri tavalla:
Minuuttien kohdalla ((minuutit + sekunnit / 60f) / 60f) * -360f //(60f koska kellotaulussa 60 minuuttia)
Tuntien kohdalla ((tunnit + minuutit / 60f) / 12f) * -360f //(12f koska kellotaulussa 12 tuntia)

Digikello

Käytin digikelloa varten TextMeshPro:ta, joka antaa enemmän tekstivaihtoehtoja. Siirsin syntyneen kanvaasin Kellon lapseksi, muutin kanvaksen renderöinnin ”World Space”-moodiin, jotta digikellon saisi järjestettyä viisareiden taakse ja lopuksi mitoitin kanvaksen sopimaan kellotauluun, jotta se skaalautuisi oikein eri resoluutioilla.

Digikellon tekeminen on suhteellisen helppoa, sillä tarvitsee vain asettaa tekstikenttään aikayksiköiden väliin ”:”-merkki, nolla (jos luku on yksilukuinen) ja tietenkin oikeassa järjestyksessä. Tämä onnistuu tekemällä kolme uutta string-tyyppistä muutujaa (sDigi, mDigi ja tDigi) ja asettamalla niihin oikeat aikayksiköt esim: aika.Second.ToString(”00”), jossa ”aika.Second” on mahdollista pyöristää, jos haluaa. Loin uuden peliobjekti-muuttujan tekstikenttää varten (digikello), jotta saan tekstikenttään sijoitettua:

digikello.text = tDigi + ”:” + mDigi + ”:” + sDigi;

jonka jälkeen analoginen ja digitaalinen kello toimivat.

Muita ominaisuuksia:

Millisekunnit

Loin metodin ”MsToggle” ja toggle-napin, joka päällä ollessaan asettaa mss-muttujan todeksi ja lisää msDigi-muuttujan digikelloon. Jos mss on epätosi, niin msDigi on tyhjä merkkijono.

Stop-nappi

Loin ”Sammuta”-metodin ja stopnappi-togglen, joka muuttaa ehtolauseen, joka sisältää koko kellon toiminnan, epätodeksi.

Oma aika -tekstikenttä ja sille Error-viesti

Loin InputFieldin, OmaAika metodin ja Error-viestin. Kun käyttäjä laittaa InputFieldiin sopivan ajan muodossa: 00:00:00, kello pysähtyy (stop)nappi.IsOn = true avulla ja scripti erottaa ”:” kohdissa numerot toisistaan Split() funktiota käyttämällä ja asettaa ne listaan, josta numerot laitetaan digikelloon ja muunnetaan asteiksi viisareita varten. Jos käyttäjä laittaa jotain muuta tai liikaa, tulee punainen virheteksti-objekti aktiiviseksi, joka lähtee kun painaa stopin pois päältä. Virheitä seurataan Try ja Catchin avulla.